

UNIVERSITY OF TARTU  
Institute of Computer Science  
Computer Science Curriculum

**Magnar Aruoja**

**Tarakas – a cross-platform piggy bank application for children**

**Bachelor's Thesis (9 ECTS)**

Supervisors: Pelle Jakovits, PhD  
Kristjan Kool, AS LHV Pank

Tartu 2017

## **Tarakas – a cross-platform piggy bank application for children**

### **Abstract:**

Even though we live in a materialistic world, financial education is not part of our everyday life. Knowledge not acquired during childhood is harder to be gained later on. This bachelor's thesis describes cross-platform money collecting and planning mobile application development for kids, which can be integrated with real bank systems in the future. There are money (allowance) planning applications in the market, but they have disadvantages which may be blocking their wider spread. The author has pinpointed core issues within banking applications, which prohibit their effective use by children. A child-friendly piggy bank application prototype has been developed to deal with those issues. Children can define their goals (dreams) and plan money to achieve those. In addition to that, the author brings out future work that can be done to deploy the application to production for end-users. Prototype application's technical solution is based on state of the art technologies, which were chosen from many to match author's current skill base without compromising quality.

### **Keywords:**

Money, piggy bank, education, cross-platform, mobile application, Ionic Framework, Angular 2, children, smartphones, financial literacy, TypeScript

**CERCS: P175 (Informatics, systems theory)**

## **Tarakas – platvormiülene rahakassa rakendus lastele**

### **Lühikokkuvõte:**

Olgugi, et elame materjalistikus maailmas, pööratakse finantsharidusele liiga vähe tähelepanu. See, mis jääb noores eas omandamata, on hiljem kordades keerulisem järele aidata. Käesolevas bakalaureusetöös kirjeldatakse platvormiülse raha kogumise ja planeerimise mobiilirakenduse loomist lastele, mida on tulevikus võimalik integreerida pangasüsteemiga. Lastele suunatud (tasku)raha planeerimise rakendusi maailmas küll eksisteerib, aga neil esineb puudusi, mis on saanud takistuseks nende laiemale levikule. Autor on oma töös välja toonud peamised probleemid, mille lahendamiseks on valminud lapsesõbralik rahakassa lahenduse töötav prototüüp. Selle abil on võimalik lapsel püstitada eesmärged (unistusi) ning planeerida raha nende täitmiseks. Lisaks koostab autor nimekirja täiendavatest töödest, mille arendamise järel on rakendus lõppkasutaja jaoks valmis. Valminud rakenduse tehniline lahendus põhineb uusimatel tehnoloogiatel, mille valik põhines arendusmugavuses ning samas järele andmata kvaliteedis.

### **Võtmesõnad:**

Raha, rahakassa, haridus, mitmeplatvormilisus, mobiilirakendus, Ionic raamistik, Angular 2, lapsed, nutitelefonid, finantsharidus, TypeScript

**CERCS: P175 (Informaatika, süsteemiteooria)**

## Table of Contents

1	Introduction.....	4
1.1	The problem.....	4
1.2	The solution .....	4
1.3	The motivation.....	5
1.4	Outline.....	5
2	Background.....	6
2.1	Children and money.....	6
2.2	Cross-platform development.....	6
2.3	State of Art.....	7
3	Related work .....	10
3.1	Bankaroo.....	10
3.2	iAllowance.....	11
3.3	Yuby.....	11
3.4	RoosterMoney.....	12
3.5	Conclusion .....	13
4	The solution .....	15
4.1	The client-side Ionic application.....	15
4.2	The backend Spring Boot application.....	22
4.3	Proposed solution to integrate with real bank account .....	23
5	Feedback .....	27
6	Conclusions.....	29
7	Future work.....	30
8	References.....	31
	Appendix.....	32
I.	Source code.....	32
II.	Android vs iOS user interface.....	33
III.	License .....	35

# 1 Introduction

## 1.1 The problem

Today's technology is growing at rapid rate, new hardware and software is built every day. It is good that people can easily adapt to new technologies and make use of them. Children are no different, actually it is the other way around – children can learn anything new at ease while older people struggle much more to use smartphones, for example.

Even though technology has helped us to solve many problems and to make everyday communication easier, there are many fields that still need improvement. One of those is financial literacy. It is a fact that 4% of Estonia's population (around 50 000) are in debt because of term loans and around 10% (130 000) of people in total have taken term loan as of 1<sup>st</sup> of January 2014 [1]. It shows that people do not really plan their money or save it for the future. With no financial backup, people start taking loans when unexpected problems occur. To improve the financial knowledge of people, we should educate them from the very beginning – starting from our kids to save and spend wisely, so they do not have loan money.

Parents struggle to teach kids the value of money and how they should spend or collect money. As the world is doing more and more virtual transactions (using credit cards, making online payments), kids understand less about "real money." Money is just something abstract for them and they cannot really tell the value of 20 euros, for example.

## 1.2 The solution

Although there are some really great piggy bank applications that try to solve this problem for both children and their parents, they still lack some key features or they are too hard to use for small kids. Most of them have too many features which makes it quite hard for children to take the most out of it. Furthermore, they still represent data only in numbers and percentages which does not make much sense for grade-schoolers<sup>1</sup>, for example.

Author's challenge is to create a mobile application to address the problem with state of the art technology. One way to achieve this, is to make children create goals for themselves and then representing the money and collection progress through colours and real money bills. This will give children better idea how much money different items cost and that money is not just a number.

The purpose of this thesis is to develop an application that helps children to learn (and parents to teach them) the value of money and give children better financial education. The application should be supported for 2 most popular mobile platforms – iOS and Android.

As the application's main functionality is not really platform dependant, then the best way to cover the main functionality is to create an application using cross-platform technologies. The application should cover these main aspects to make financial education easier for child:

- can create or delete goals to collect money towards,
- can add or remove money from selected goal,
- can take real photo of goal using smartphone's camera,
- can see overview of all goals, collected money and unused money,
- can see detailed overview of specific goal,
- simple (special) user interface customized for children to help illustrate value of money and how much different things cost.

---

<sup>1</sup> 7 to 12-year-old children

Additional functionalities that will be covered:

- proposed solution to integrate with real bank account,
- prototype solution to collect money through link (for example if child has birthday then people can send money-gift straight to one's bank account).

Even though cross-platform frameworks usually give possibility to develop same application for more different platforms than just iOS and Android, the purpose of this thesis is to concentrate on Android and iOS, and develop a working prototype – rather than a fully working ready to deploy solution.

### **1.3 The motivation**

The author of this thesis is eager to learn new mobile development technologies and thus is ready to try and develop something new using cross-platform hybrid mobile application technologies. In addition to that, this would also help other people and especially children. The author would love to see his work being used in real world systems so integration with real bank is the key to this.

### **1.4 Outline**

Chapter 2 describes the background of this Bachelor's thesis, shows some real-world examples that the problem exists and gives an overview of technologies, frameworks and approaches that will be used to solve the problem.

Chapter 3 focuses on similar applications, brings out their advantages and disadvantages and shows what will be different in author's solution.

Chapter 4 is a detailed overview of author's approach solving the problem. There is a detailed review of the application main usage with screenshots of final application.

Chapter 5 describes test results and feedback from using the application.

Chapter 6 concludes the work author has done and brings out main ideas, advantages and disadvantages of this thesis

Chapter 7 discusses about suggestions for future work

## 2 Background

### 2.1 Children and money

We live in a materialistic world where money is one of the most important assets. Unfortunately, children do not tend to understand the value of money. The main cause is that they have never really been taught this. Financial education is not part of our everyday school. As many authorities suggest – children need to learn “money does not grow on trees” – they need to tell difference between their *wants* and *needs*.

Laura Shin published an article [2] in Forbes: “The 5 Most Important Money Lessons to Teach Your Kids.” She wrote that money can be a result of hard work and sometimes one has to save money to buy bigger things. In addition to that, she insists that parents should teach their kids how to save money by creating goals. Kids should learn the difference between saving and spending. As children get older, parents can introduce new activities to help them understand the world and money better.

Steven Richmond adds in his Investopedia article [3] that parents should be the role models for their kids. As children will most likely pick everything up from their parents then children should be taught what is wrong and what is right. What is more, he emphasizes that children should also learn by playing games such as *Monopoly* or *Sim City*.

Liliia deCos [4] did research on “Opportunity for banks to utilize gamification as a tool to promote financial education to children.” She gathered data from different people by interviewing them and sent out questionnaires. DeCos concluded that financial literacy is very beneficial and very difficult for parents at the same time. Parents do not know much how they should teach children about money but they are willing to learn and get help from different applications or games that try to help children.

In addition to that, she points out that the competition in financial education market is low, thus banks should provide applications for children so both kids and banks themselves could benefit from it. Banks could increase their existing customer base and gain better reputation.

### 2.2 Cross-platform development

It is a known fact that the world is going more and more mobile. There is an application for almost anything. Ritesh Patil [5] points out that many organizations have started to build mobile applications for their services or products. Different kind of customers use different devices so it is important to make sure that all customers can use the application. He adds that for companies or development teams this means that they need to provide an application for almost each popular platform which means that mostly they need to develop the application for both iOS and Android.

This brings developers into a situation to choose whether to first build an application for iOS or Android. This means that most of the time twice the same amount of time and work is needed to build an app for both platforms as iOS and Android native code cannot be shared.

Cross-platform development technologies simplify the development cycle. Basically, this means that developers code the business logic and app specific rules once and deploy it anywhere. Sometimes different platform specific styling can be done to give application more native feel and look. Patil focuses on two most popular approaches for cross-platform mobile development – native cross-platform and hybrid (HTML5) cross-platform applications.

## Native cross-platform applications

Patil [5] explains this approach as some third-party vendor creates unified API<sup>2</sup> that can be used to develop an application (using the programming language the API is written in) for multiple platforms. He adds, that usually these frameworks are bundled with tools that make deployment easy for any platform.

Compared to the hybrid application development, native cross-platform development frameworks usually do not provide platform independent user interface (UI) frameworks which means the developer has to write separate UI for both platforms, for example *Xamarin*<sup>3</sup> has separate *iOS Designer* feature and different Android specific layouts. Even though the development process might be slower, the application's response and feel should be a little bit smoother compared to hybrid application development.

## Hybrid 'HTML5' cross-platform applications

Hybrid applications are so called half-native apps. Patil [5] describes this approach as a web application that actually runs in native application's web container (*WebView* component). As they are wrapped in native app they can access different native features such as camera using JavaScript API provided by the hybrid framework.

Compared to the native application development, hybrid cross-platform application development frameworks also provide cross-platform UI frameworks which makes it very easy for developer to support both platforms native-feel UI (such as menus and navigation buttons). Another upside of using hybrid application frameworks is that any kind of JavaScript library can be easily imported and developers can make use of CSS and JavaScript together to create neat user interface for both platforms with ease. Of course, one has to consider that native UI can sometimes be faster when there is some really complex view to render.

## 2.3 State of Art

### Ionic 2 Framework

Ionic [6] is an open-source cross-platform framework built on top of Angular<sup>4</sup> 2 (*TypeScript*<sup>5</sup>) and Apache Cordova. It can be used to build hybrid mobile applications which means the applications will have support for native device features such as Bluetooth, Fingerprint Authentication and camera, for example. Ionic application runs in native application's web container. The framework supports iOS 7+, Android 4.1+ and Windows Phone operating systems [7].

The author will be using Ionic 2 framework in his application because Ionic is full hybrid cross-platform framework which means it provides cross-platform UI components that make the user interface development fairly easy. Also, because the author already has some experience developing web applications, he is eager to learn new web development technologies such as Angular 2 (on top of *TypeScript*). There are potential alternatives, such as Google's Material Design<sup>6</sup> and Polymer 2.0<sup>7</sup>. First concept mostly focuses on Android UI and latter is not specially designed for mobile applications. Author decided to stick with

---

<sup>2</sup> Application Programming Interface

<sup>3</sup> <https://www.xamarin.com/>

<sup>4</sup> <https://angular.io/>

<sup>5</sup> Programming language - <https://www.typescriptlang.org/>

<sup>6</sup> <https://material.angularjs.org/latest/>

<sup>7</sup> <https://www.polymer-project.org/>

Ionic because it has native specific UI for both platforms and it has larger community than others.

## Spring Boot and Security

Spring [8] is an open-source Java framework that makes easy to build simple JVM-based applications. Spring is mostly used for backend development and they can be run anywhere JVM can.

Spring Boot [9] is a Spring project which helps developer to set up a lightweight, stand-alone application with minimum set-up effort. It contains some third-party libraries that are widely used for web development and it helps to automatically configure the project. Also, it provides many production-ready features such as metrics.

Spring Security [10] is another open-source Spring project that has highly customizable authentication features. Authors of this framework say that this is the de-facto standard for securing Spring-based applications. It makes easy to authenticate and authorize requests and protect against different attacks, for example CSRF<sup>8</sup>.

Because of the Spring Boot's built-in features and author's experience in Java, this is the backend technology that author will use for the application. Actually, there are many alternatives to choose from – whether to use Python language and *Django* framework or PHP language and *Laravel* or *CodeIgniter* framework and many more.

## Mongo Database

MongoDB [11] (shorthand for Mongo Database) is a NoSQL database that stores data using JSON<sup>9</sup>-like documents. Data types for these documents also include arrays and binary data. MongoDB makes data mapping so easy that usually there is no need for an ORM<sup>10</sup> layer. Apart from other NoSQL databases, MongoDB provides secondary indexes and in addition to that it has many security capabilities. The database offers different solutions to analyse data (search, in-database analytics, cross-document relations and much more).

In addition to that, there are Mongo database drivers for almost all popular programming languages, needless to say Spring Boot comes with MongoDB repository drivers which makes it very easy to access the database from Java application.

As the Mongo database can store byte array in its documents, it makes it easy for developer to store small images in database as well as any other user specific data. And as the application implementation does not require data to be relational in SQL way the NoSQL database is the easiest way to store data for author.

Of course, there are many alternatives to store data starting from SQL databases such as MySQL or PostgreSQL to NoSQL alternatives such as Hadoop or Cassandra, but the author's reason to use Mongo is its simple integration with Spring Boot.

## Security

It has become quite a standard that applications are being developed frontend and backend separately. This means that different teams can develop different parts of applications simultaneously to release new products faster. But separating the frontend and backend makes it a little bit harder to secure the whole application. Today's growing standard is to secure

---

<sup>8</sup> Cross-Site Request Forgery - [https://www.owasp.org/index.php/Cross-Site\\_Request\\_Forgery\\_\(CSRF\)](https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF))

<sup>9</sup> JavaScript Object Notation

<sup>10</sup> Object-relational mapping



applications using token-based authentication. This means, that first time user authenticates against server, a token is returned. Now, on each following user request, the token is required to identify the person who is making requests. In addition to that, these applications do not need sessions anymore. Token-based authentication allows to do stateless requests without the need of keeping track of current active sessions.

### 3 Related work

The most popular money saving apps for kids are reviewed by Juan Castillo in *NerdWallet*'s article<sup>11</sup> "*Mobile Money Apps: Tech Helps Teach Kids to Save.*" Based on the article, author chose three most popular applications and added another by his own research. At first each application's advantages and disadvantages are described based on the article, products' respective websites and application itself. Finally, results are compared against thesis author's solution.

#### 3.1 Bankaroo

*Bankaroo*<sup>12</sup> is more or less a financial education platform. It has *Bankaroo for Schools* brand that is meant to be used by schools and teachers to teach kids math and financial education. The iOS app UI shows that the app is developed using some older cross-platform framework because on iPhone 6 one can clearly see after opening the app phone status bar (clock and battery info) is different than the current iOS native one.

- Advantages: cartoon-like UI for kids; list of upcoming activities (incomes and out-comes that are projected in the future)
- Disadvantages: outdated; in author's opinion, bad native feel because of old cross-platform technologies being used; crowded with too much functionality

Figure 1 shows *Bankaroo*'s main screen, where user can see total money, all goals, badges and upcoming activities.

Figure 2 shows new goal creation in *Bankaroo*. User can set name, price and icon for goal. Also, initial collected amount can be added.

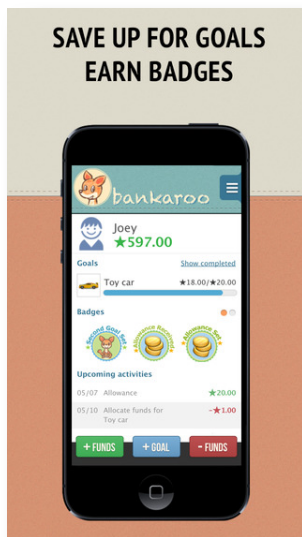


Figure 1. *Bankaroo*'s main screen<sup>13</sup>

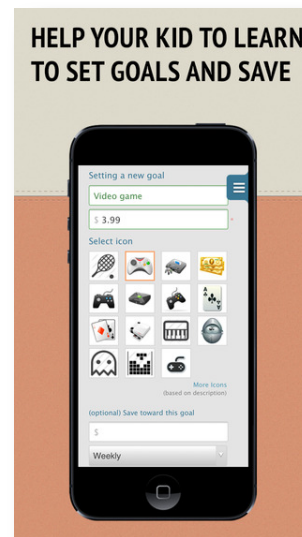


Figure 2. *Bankaroo*'s new goal set up screen<sup>13</sup>

<sup>11</sup> <https://www.nerdwallet.com/blog/banking/mobile-money-apps-kids/>

<sup>12</sup> <https://www.bankaroo.com/>

<sup>13</sup> <https://itunes.apple.com/us/app/bankaroo-virtual-bank-for/id504924470?ls=1&mt=8>

### 3.2 iAllowance

*iAllowance*<sup>14</sup> is another great app to teach kids how to save money. As it has quite many features the UI can get a little bit hard for smaller kids to understand everything that is going on in app. Parents can give their children specific tasks to complete (chores) so kids can earn money. The real disadvantages for this app are lack of Android support and it is not free. Author's opinion is that the UI should be a little more simplistic for kids to use the app.

- Advantages: push notifications for chores for reminding children
- Disadvantages: it is not free (\$3.99), only for iOS

Figure 3 shows *iAllowance*'s "Rewards" screen. Rewards have progress bar showing current progress with also colour changing dynamically.

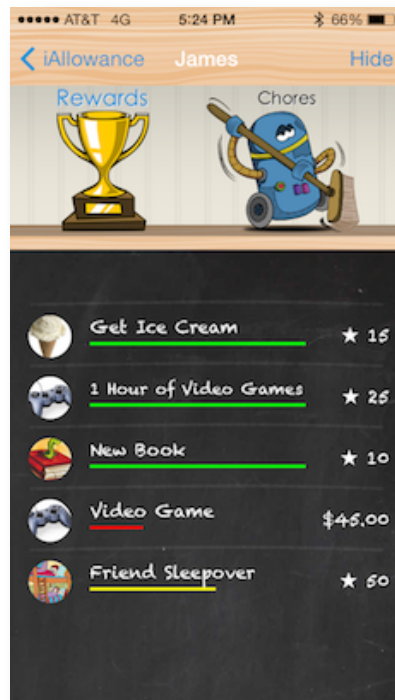


Figure 3. *iAllowance* rewards screen<sup>15</sup>

### 3.3 Yuby

*Yuby*<sup>16</sup> is a virtual money saving app from Union Bank. This app is the closest to meet the requirements author has - it is simple, has just main features (create goals and collect money towards them), easy to use. Union Bank could also make possible to bind real accounts with the app so children have real overview of money.

- Advantages: simplistic UI, for kids aged 6-11; special tasks for kids to earn money

<sup>14</sup> <http://www.jumpgapsoftware.com/allowance/index.html>

<sup>15</sup> <http://www.jumpgapsoftware.com/iallowance.html>

<sup>16</sup> <https://www.unionbank.com/campaigns/yuby/>

- Disadvantages: owned by real bank, but is still not integrated with real account balance

Figure 4 shows detail screen of user's wish progress. Money is shown in packages, where each bill is worth one euro. In addition to that, price of wish, current money and still needed amount are shown.

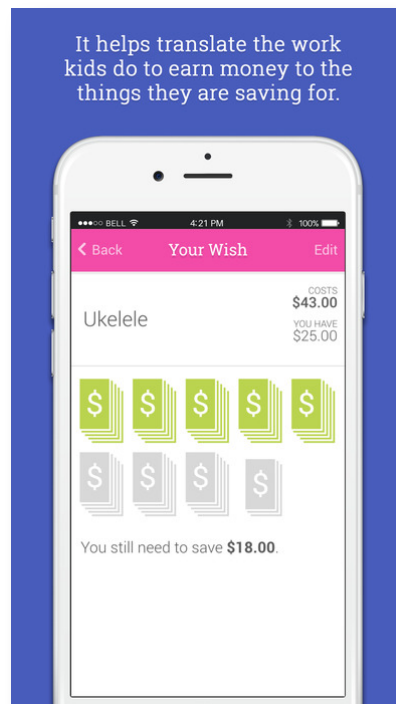


Figure 4. *Yuby's* "Your wish" screen<sup>17</sup>

### 3.4 RoosterMoney

*RoosterMoney*<sup>18</sup> is another great money saving application for Android and iOS. It has really good looking UI, simplicity and easy to use. Maybe the progress bar showing collected percentage of target should be something more than just bar with numeric value. For children, it would be easier to understand if there are more graphical elements that represent how much is collected and how much is yet to be collected.

- Advantages: great looking UI, simple to use, elegant
- Disadvantages: graphical representation of current progress

Figure 5 shows *RoosterMoney's* "Goals" screen. All user goals and total amount added to goals are shown there. User can also add new goal from this screen.

<sup>17</sup> <https://itunes.apple.com/us/app/yuby/id783848298?mt=8>

<sup>18</sup> <https://www.roostermoney.com/>

Figure 6 shows *RoosterMoney*'s "Boost child" screen. Parents can use this screen to give child money (boost).

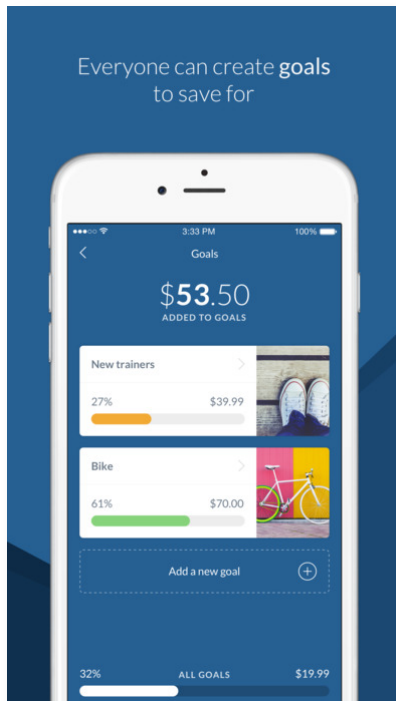


Figure 5. *RoosterMoney*'s "Goals" screen<sup>19</sup>

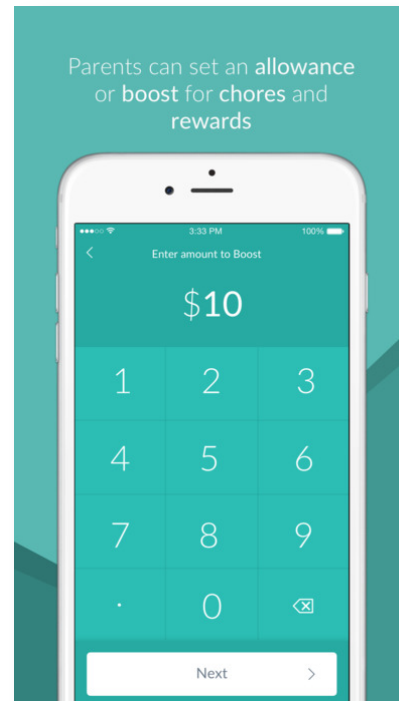


Figure 6. *RoosterMoney*'s "Boost child" screen<sup>19</sup>

### 3.5 Conclusion

There are many piggy bank solutions available in the market, but there is still room for innovation. Most of the existing applications seem to have different focus compared to the purpose of author's application – concentrating on chores and allowance rather than helping children to make reasonable decisions with money. Also, the applications are overwhelmed with "fancy" features which may make the application hard to use for children. Even though, gamification features in general are a plus for this target group, it should not distract children from the main idea.

Another gap found is lack of integration with real bank accounts and applications. In a situation where majority of adults are using online banking, it is probably not wise to keep children in virtual sandbox, which makes it harder to enter the real world and make financial decisions when they are getting older.

Almost all of them have usability issues and they really do not help children understand the value of money. The data is mostly represented only in numbers and percentages. The author of this thesis tries to solve this problem by adding more visual components to the application.

<sup>19</sup> <https://itunes.apple.com/us/app/roostermoney-manage-allowance-and-track-chores/id743769885?mt=8>

Another extra feature that author added to his app is that children can ask money for specific goals, for example on birthday. Using the application, child can send out birthday invitation to all guests with his or her wish (specific goal) and bank account number which is connected to the app. It will save parents from 10 little plastic (i.e. 10 guests with *circa* 10 euros worth presents each) toys and will help the child to receive one real present (worth for example 100 euros) that he or she has been dreaming about (and collected money for).

In addition, there are no piggy bank apps in Estonian so Estonian children cannot take advantages of current applications.

## 4 The solution

This chapter describes the architecture of author's solution. It gives an overview of client-side application which was developed using Ionic Framework and its capabilities, overview of back-end Spring Boot application and also describes author's vision for integrating the created application with a real bank account.

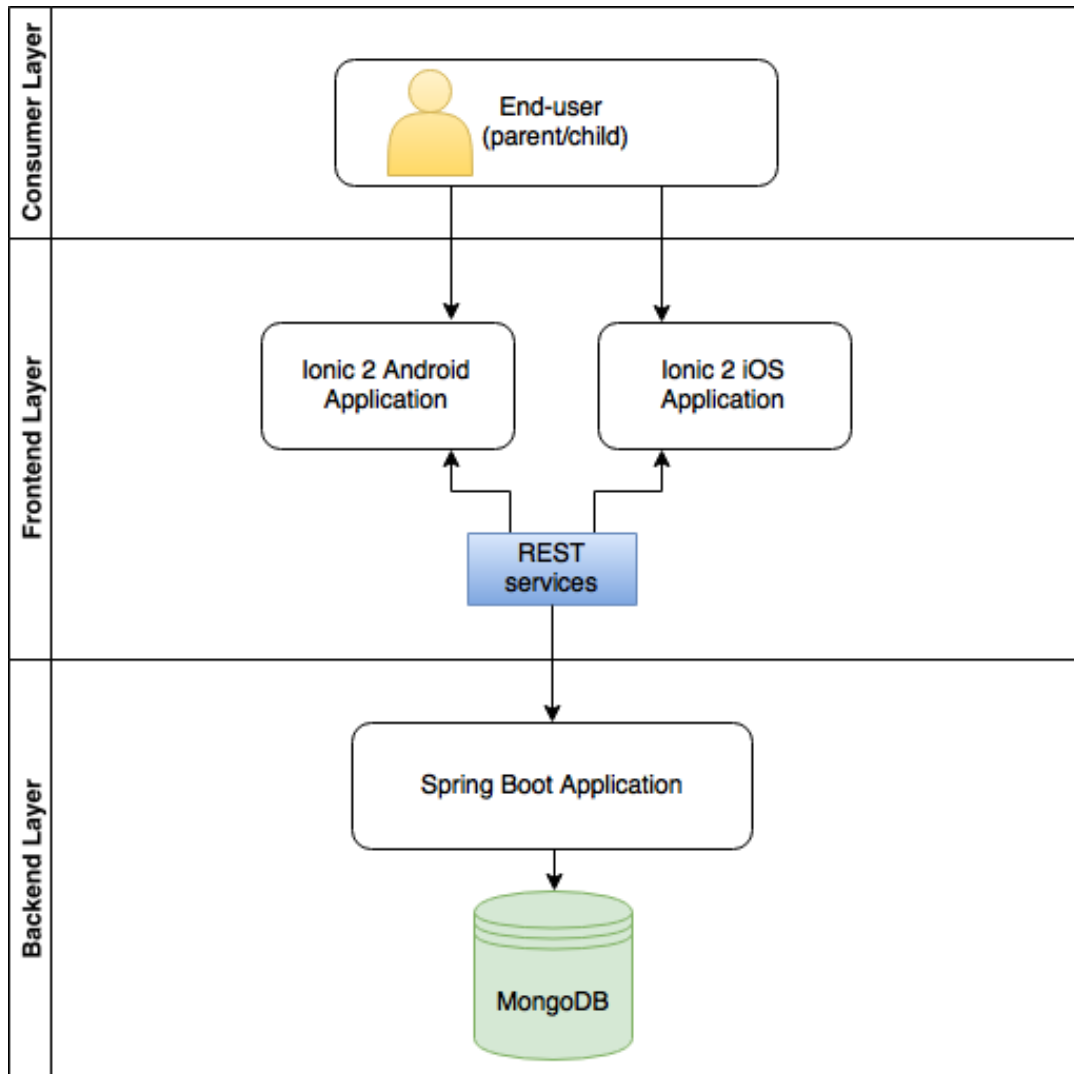


Figure 7. A simplified overview of application's architecture

Figure 7 shows main components of the application. An end-user can choose between iOS or Android smartphone to use the application. The Ionic application exchanges data with backend through different *REST*-services. Backend layer consists of two main components – a Spring Boot application and Mongo database. The backend application is connected with Mongo database through Mongo repository Java drivers. All user data is stored in the MongoDB.

### 4.1 The client-side Ionic application

This subchapter will give an overview of the application's user interface and main functionalities.

## Login and registration

Figure 8 shows “Login” screen. This is the initial screen of application. If a user already has an account, then the user can authenticate itself by filling username and password, otherwise new account needs to be created.

The Ionic application allows new users to register a new demo account. Demo account is an account which is not connected to any real bank account. User can add to or remove money from demo account anytime.

Registration of the new demo account is as simple as logging in because user only has to provide the username and password.

Once registration is successful, user will be automatically logged in and JWT<sup>20</sup> is saved to in-app memory. JWT is basically a digital signature that holds user specific information and can be verified on each request<sup>21</sup>. Next time when the application is opened, user will be automatically logged in and can use the application straight away.

JWT will be deleted when application’s cache is cleared.

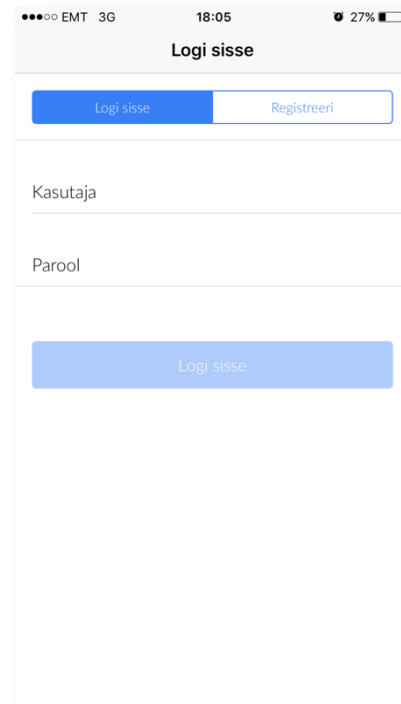


Figure 8. Tarakas “Login” screen

---

<sup>20</sup> JSON Web Token

<sup>21</sup> <https://jwt.io/introduction/>



## Main screen

After authentication, user is shown the application main screen. The main screen consists of three parts – upper navigation bar, main content and tabs bar in the bottom of screen. In the beginning, main screen is empty as on Figure 9 can be seen.

Navigation bar has “+” button that can be used to create a new goal. The main content gives an overview of user’s current balance – how much money can be spent and how much money is already moved to certain goals and how much money user has in total.

There is also a message panel that notifies user if he has some spare money left or if user’s balance has changed and some money needs to be taken out from any of the goals below (negative spare money balance).

On Figure 10 goals are shown as cards. On the left side of the card, goal name, collected amount and price are shown. On the right side of the card user can see the picture of goal (if any was added on goal creation). Cards have dynamic background – it means that the more user has collected money for certain goal, the greener the background will get indicating that the goal can be soon achieved.

With this kind of approach, a child will have better overview of goals and it is easier to make difference which goal can be completed earlier. The author of this thesis thinks that this is much easier for children to tell difference than seeing percentages of collected amount.

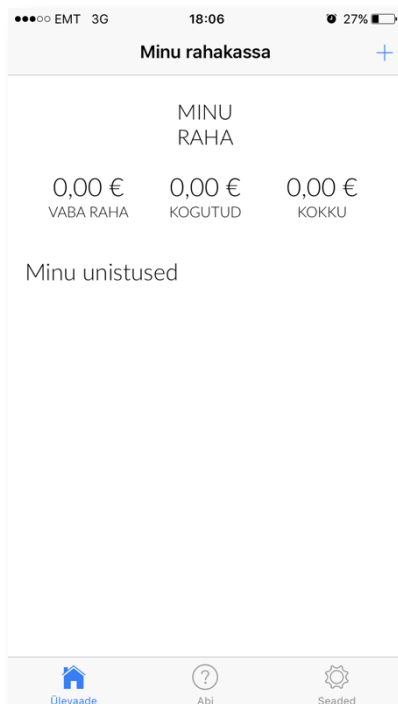


Figure 9. Screenshot of empty main screen



Figure 10. Screenshot of main screen with added goals

## Adding new goals

To add new goal, user can open modal screen by pressing “+” from the main screen navigation bar. Adding new goal is made as simple as possible. Figure 11 shows empty goal creation screen. User can enter the name and the price of the goal and also take a picture from the object using phone’s camera.

Once all fields are filled, user just needs to click “Add” button and new goal will be added. Result can be seen right away from the main screen. If user does not want to add the goal, “Cancel” button can be clicked from the navigation bar and the modal window will be dismissed with no data being saved. Figure 12 shows an example of new goal creation with all fields filled.

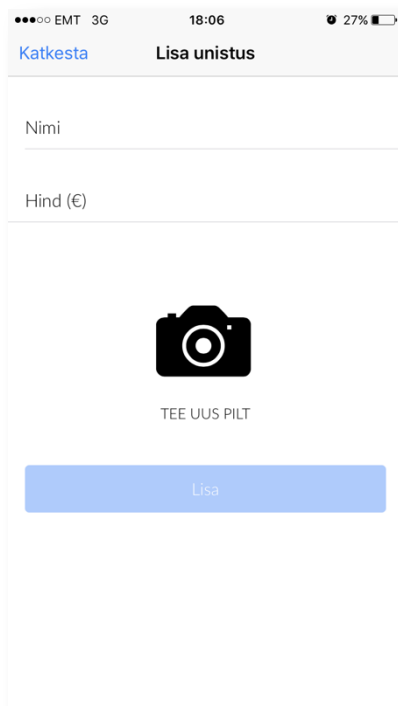


Figure 11. Screenshot of empty goal creation screen

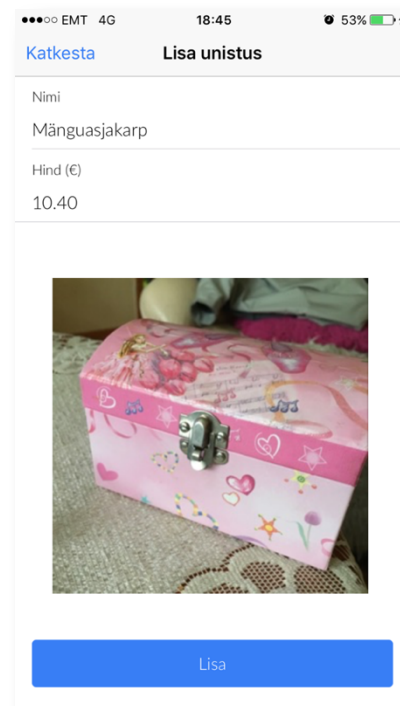


Figure 12. Screenshot of new goal creation with filled fields

## Goal details

The goal details screen gives an overview of selected goal. It uses similar cards as the main part of the user interface. At the top, there is “Spare money” (“VABA RAHA”) label showing the amount of money that can be spent. The “Collected” (“KOGUTUD”) and “Need to Collect” (“VEEL VAJA”) cards’ background colour is changing dynamically.

Background colour of the collected amount changes according to the percentage of total amount and the spare money background changes according to how much money is yet needed to complete the goal. This means that if user has more spare money than needs to be collected, the spare money background will always be green.

In addition to that, the amount of money on both cards are shown using correct coins and bills. This approach will help children to understand the value of money a little bit better than just numbers. The number in the little red circle shows up if more than one coin or bill of the same type is needed or collected. Figures 13 and 14 show different colour combinations of different cards depending on how much amount is needed and how much is already collected, also, how much spare money user has.

This will give children better visual overview and help them to think about whether to add money for this goal or not and whether they have enough money to complete the goal.

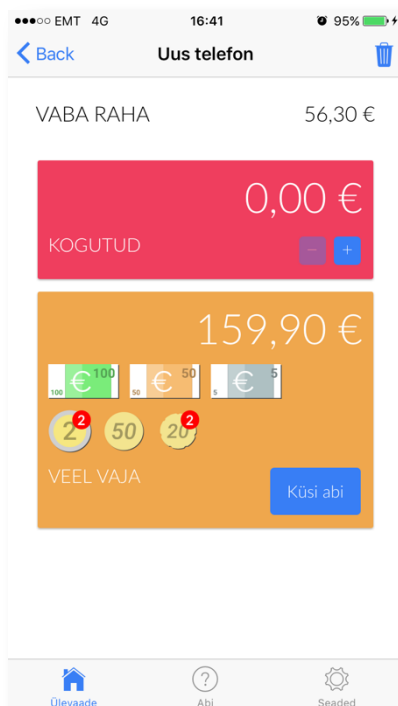


Figure 13. Screenshot of goal details screen – nothing yet collected

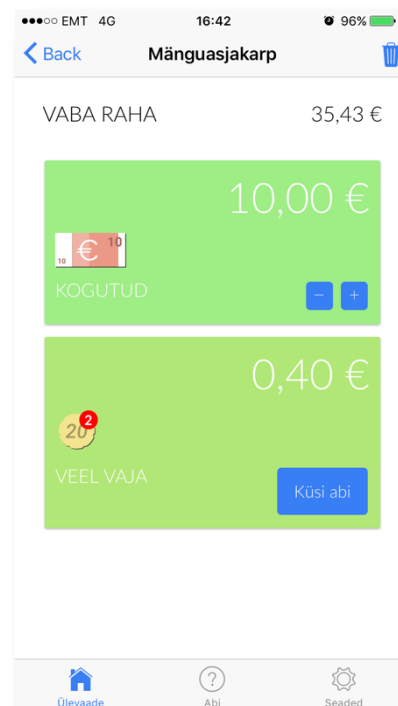


Figure 14. Screenshot of goal details screen – goal almost completed

The “*Need to Collect*” card has “*Ask for Help*”<sup>22</sup> button which can be clicked to ask for help on achieving the goal. If user clicks the button, 2 choices will appear – whether user wants to collect money for this goal as his or her birthday gift or it is more like collecting money for bigger project. Choosing any of these will open phone’s default e-mail application and subject and default message of selected goal is already provided. User only needs to add recipients to who the e-mail should be sent to. The template will include child’s bank account information so people can help the child by making a transaction to his or her account.

The “*Collected money*” card has two buttons, “+” and “-“, representing the actions to add money for a goal or take money off the goal. This feature also gives the child a little bit “entertainment” – a child can put money in or take out and see whether the colour change drastically or not – especially for those younger kids who are not financially educated yet. By pressing any of those, an alert window will pop up with options to choose from how much many can be added or removed.

In the navigation bar – on the right-hand side – there is a garbage can icon representing the “delete” action. By tapping this, user will be prompted whether user really wants to delete the goal or not. If “Yes” is pressed, the goal will be deleted.

---

<sup>22</sup> “*Küsi abi*” button

## Help screen

The “Help” screen contains helpful information about money bills – visually showing how much money each next coin or bill is worth. This screen can be reached from the bottom tabs bar (by pressing “*Abi*”) anytime when no modal screen is opened. Figures 15 and 16 show how different coins and bills are represented and combined, to make the value of different bills and coins a little bit clearer for child.

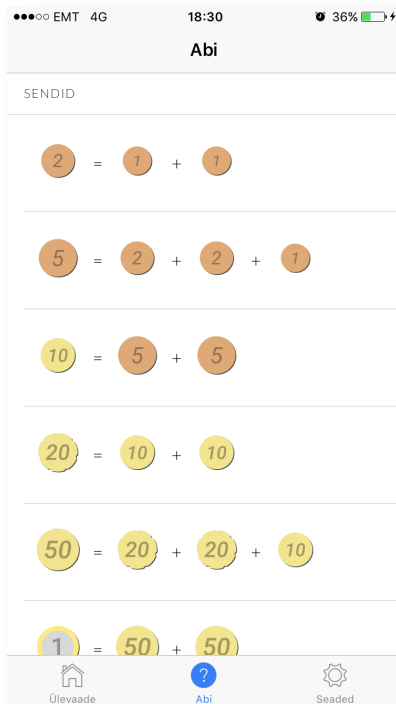


Figure 15. Screenshot of coins in the help screen



Figure 16. Screenshot of bills in the help screen

## Settings screen

The “Settings” screen has currently only 2 options – “Log out” and “Add money” (Figure 17). The “Add money” option is available for demo accounts only, so user can add money to account. This feature is meant for children so they can see what they can do with money and how the application functionality should help one. Also, this “Add money” functionality can be used to remove money from account by just adding negative amount of money. This way it can be imitated like real transaction was made from account and now the child has to remove money from any of his or her goals to get the balance back to zero or positive. Figure 18 shows how money can be added to account.

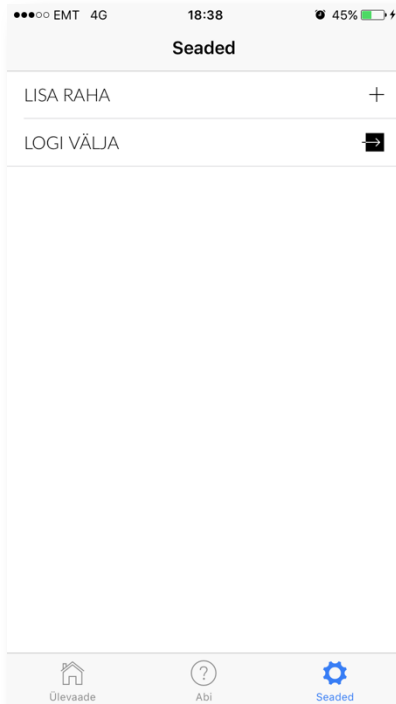


Figure 17. Screenshot of settings screen

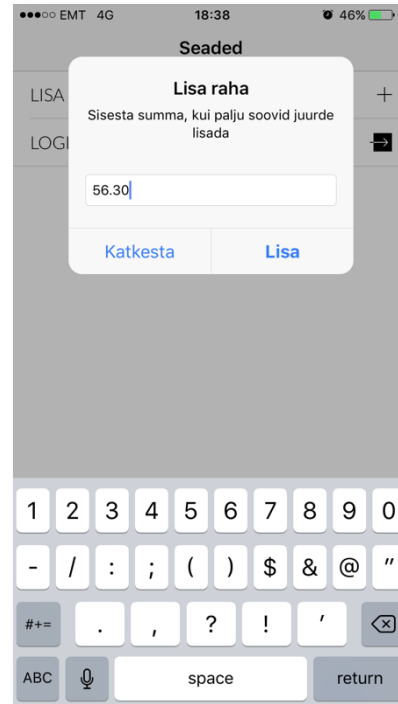


Figure 18. Screenshot of user adding money to demo account

## 4.2 The backend Spring Boot application

The application’s backend consists of two main parts the server-side Spring application and Mongo database (Figure 7). Both instances are deployed to the virtual Ubuntu server, hosted by Vultr<sup>23</sup>. The virtual server specifications are: 1 CPU, 1024 MB of RAM, 25 GB SSD and monthly bandwidth of 1000 GB. The operating system is 64-bit Ubuntu 17.04.

The Spring application is built using RESTful API guidelines<sup>24</sup>. This means that application has multiple controllers that can be used to manipulate resources – for example GET request against specific URL will return data and POST against the same URL should insert new or

<sup>23</sup> <https://www.vultr.com>

<sup>24</sup> <https://github.com/Microsoft/api-guidelines/blob/vNext/Guidelines.md#71-url-structure>

update existing entry in the collection. Every controller's endpoint uses service(s) to do specific operations.

The frontend application can use 3 main services through their endpoints:

- BankAccountService
  - GET /api/balance – can be used to get user's current balance;
  - POST /api/balance – can be used to update demo account's balance by adding or subtracting the amount
- GoalService
  - GET /api/goals – can be used to retrieve goals of all users;
  - POST /api/goals – can be used to insert new goal or update existing goal;
  - DELETE /api/goals/{goalId} – can be used to delete goals, where goalId is unique goal identifier
- UserRegistrationService
  - /api/auth/register – can be used to register new user account. Every time user is created, also demo account is automatically created and connected with the user.

The Spring Boot application has also separate endpoint for authentication:

- POST /api/auth/login – returns user specific token

To make sure that users can only access to their own data, the application is secured using token-based authentication and stateless sessions. This means if user starts the frontend application, he first needs to authenticate himself using his credentials and making POST request to authentication endpoint. He will be given a JWT-token in return that he can use to make additional requests that the application needs. For this implementation, currently tokens never expire, which means that once user has authenticated himself, his token will be stored in his phone and each time he can make requests with the same token.

To make sure that endpoints are protected, there is a filter for each request that checks if token is valid or not before the service will start exchanging the data.

### **4.3 Proposed solution to integrate with real bank account**

Collecting money and learning when to spend or not is good for the beginning but dealing with the virtual money can get boring. To make the process real, integration with real bank account is necessary. Currently as known for author, no piggy bank application for children supports real bank integration. This could be an innovative feature that could attract children full attention. As of now, author's conceptual solution will be described because development needed for real integration is not in the scope of this thesis because of the amount of work required.

The first step to make the account integration with real bank possible is that the bank needs to develop its public API so the balance information can be requested with some kind of token, for example and web interface to create Tarakas' account.

Usually banks do not support any public APIs for different parties. In Estonia, some bigger banks have gateway systems for companies which help companies to do money management directly. But this is just an end-to-end solution for businesses - *Swedbank Gateway*<sup>25</sup>, *LHV Connect*<sup>26</sup> and *SEB Bank Interface*<sup>27</sup>.

Actually, it may be possible that in the future all European banks have public API for third-party providers. The new directive is called PSD2<sup>28</sup> – revised Payment Service Directive. The main idea is to enable bank customers to use these providers to make any kind of banking related transactions [12]. This could mean that, for example, children can buy things with the money they have collected through the same application if it is connected through the API with the real bank account.

The second step to integrate account with real bank account is that the parent makes an account for child through the web interface. Figure 19 describes how the integration between systems works. A username and password must be chosen carefully because the child will use these credentials to log in to the application. If registration data is complete then details will be sent to *Tarakas*' backend Spring application that will create the user with specified data and save the token that can be used to access real bank account balance.

If account is created successfully then child can already see his or her balance already and can start adding new goals to collect money and take the most out of the mobile application. Figure 20 describes the process between two systems.

---

<sup>25</sup> <https://www.swedbank.ee/business/d2d/ebanking/gateway?language=ENG>

<sup>26</sup> <https://www.lhv.ee/en/connect>

<sup>27</sup> <http://www.seb.ee/eng/business/everyday-banking/service-channels/bank-interface>

<sup>28</sup> <https://ec.europa.eu/info/law/payment-services-psd-2-directive-eu-2015-2366>



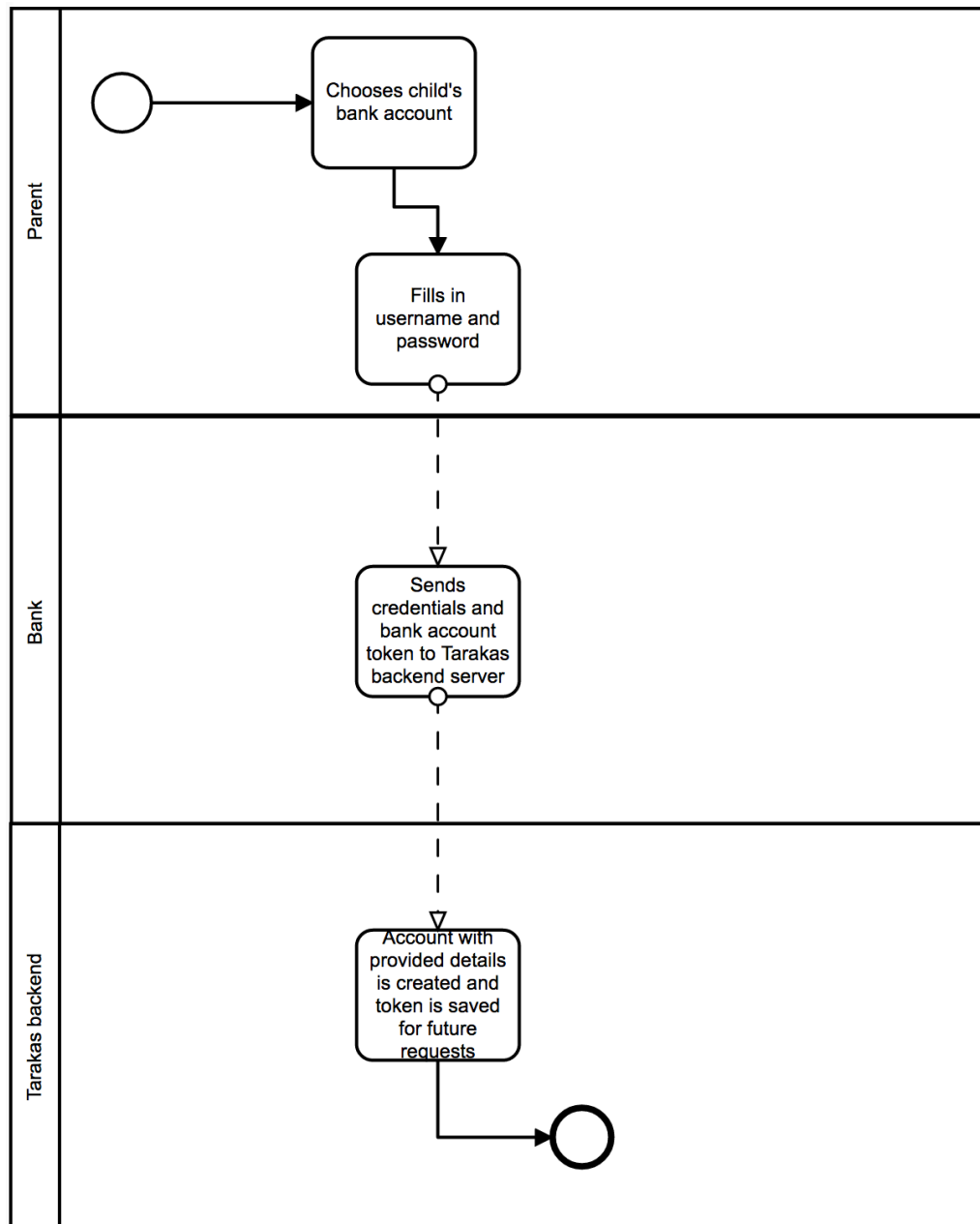


Figure 19. BPMN showing real bank account integration with Tarakas

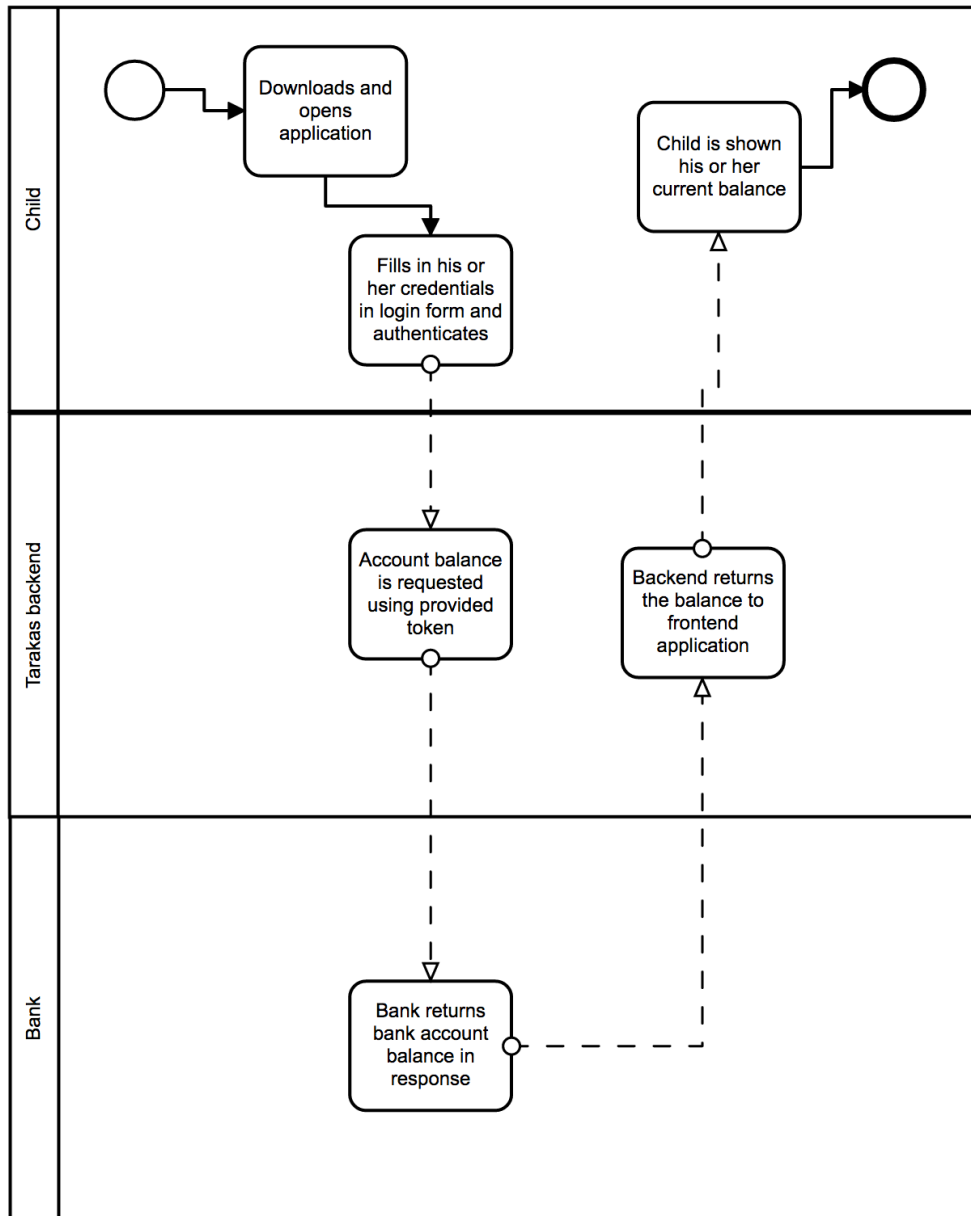


Figure 20. BPMN showing how real bank account balance is requested

## 5 Feedback

The author of this thesis presented his application and work done to children in three different schools – Peedu school (Tartu Erakool) (15 children), Miina Härma Gymnasium (25 children) and Mart Reiniku Gymnasium (24 children).

64 children in total took a part in different presentations. Children ages differed from 8 to 11 (2<sup>nd</sup> to 4<sup>th</sup> grade). Author asked children different questions about money planning and collecting, understanding the value of money and if they can tell difference between virtual money (bank account) and physical money as bills and coins.

Main questions that were asked:

- How many of you are collecting money?
- Do you use physical piggy banks or bank accounts?
- Do you collect money on certain purpose (i.e. for certain goal)?
- How do you think money can be earned?
- Have you wasted money on something you have never actually used?

Questions were asked during group chat and demoing the application to kids. Author tried to get as much information as possible from them during the talk and noted all the results.

Results showed that 89% of children (57 of 64) collect money. Almost half of the audience do it on purpose, it means they have certain things (goals) they want to buy (i.e. self-balancing scooter). Others are just collecting money for any case or because parents have told them to do so. Close to 27% of children (17 of 64) had already bank account and knew very well how virtual transactions work. Other children (47 of 64) collect their money in piggy banks and sometimes they calculate how much money they have. They told that this can be really boring sometimes and it could be better if they can see the balance right off.

Most frequent source of income for children is weekly/monthly allowance. They added that they get money for birthdays or from grandparents. Majority of children admitted that they do have spent money on things they actually do not need.

When children were asked, what can be bought with different amount of money, some of them actually understood the value very well. For example, some told that for 10 euros one can buy new earphones or mini Lego while other said that 10 candies. Some children also mentioned that they check the price from price tags and then compare things by price.

The author also asked children about how they like receiving gifts as birthday presents. 95% of children (61 of 64) wanted money because they were sure that friends or relatives do not really know what to buy as a present. They thought it is better if they can collect bigger amount of money and buy something big later.

In addition to that, children tested the application. There were 2 phones that children could use for testing, one was Samsung Galaxy S4 (Android) and the other was iPhone 6S (iOS). They liked the application as it was easy to use for them and they could see the progress of their goals. Most of the children believed that this approach can help them to collect and plan money more easily. The author was asked multiple times if the application can be downloaded from *Google's Play Store*. They also liked the idea that one can take image of the goal. One suggestion was that it would be nice to have feature to search image from Google and add it as goal's image. Author did not find any child that was unhappy or did not like the application.

According to the feedback, author thinks that the application and work done was positive. This encourages author to develop the application further and add some suggested functionalities, such as searching and adding images from web.

## 6 Conclusions

The purpose of this thesis was to develop a cross-platform piggy bank application for children to educate them financially by allowing them to create goals and collect or plan money for specific goals. The application functionality covers all main aspects – user can create and delete goals, can see goal details and add or remove money from desired goal, can take photo of goal using camera. In addition to that, to differentiate from similar applications in the market, special user interface was created to help children understand the value of money. Goal details screen contains collected and need to be collected amounts on cards in real bills and coins, and the progress can be easily seen by the background colour of these cards, whether it is greenish, yellowish or reddish.

The author also covered two additional functionalities. A solution was proposed to integrate the application with real bank system. It was found out that there is a European directive (PSD2) that may also make it easier in the future as banks need to provide public API-s in 2018.

In addition to proposed integration solution, author also implemented a prototype solution for children to ask help on their goals. For example, if it is a birthday coming up or just help needed for some project, user can send out e-mail containing all the bank account information needed where money can be sent.

The author of this thesis gave presentations on work done in different schools. He demonstrated the application to children and asked different questions about their financial education. The feedback from children was positive. Most of the children liked the idea and they wanted to use the application if possible. They believed that collecting money for something using this approach is easier as they would not spend money on just candies or ice cream.

At the end of this thesis, author describes the future work that needs to be done to deploy the application into production.

## **7 Future work**

### **Two-way authentication**

For banking application, the most important part is security. The application should have two-way authentication which means that in addition to token, also the device information will be stored into bank server. This means that if child's smartphone gets lost, the smartphone details and authentication token can be removed, so the data cannot be requested anymore from the device.

### **Goals history**

As similar applications already support goals and transactions history, it would be sensible that Tarakas application could also support this. A child will have better overview of what has been bought or deleted and the details could be analysed together with parents.

### **Edit goal details and add images from web**

It could make the user experience easier and faster, if user could edit current goal details (name, picture, price) instead of removing and creating new one. Also, it was suggested to add the feature to add images from web in addition capturing images with mobile camera.

### **Offline usage and sync**

Right now, application can be used only when connected to internet. Because children usually do not have mobile internet connection all the time (mostly they are using Wi-Fi if possible) then it would be a nice feature if the app could be used while offline. When connected to the internet, changes could be synced to server.

### **Friend list**

Today, almost every application is like a small social network. It would be nice if children could add each other as friends in the application so they can see their friends' goals and their progress as well.

### **Support for more languages**

Tarakas application currently supports only Estonian language. The application is developed while being kept in mind that in the future support for more languages can be added. For example, it would be nice to have Russian language support as many children in Estonia also speak Russian.

### **Push notifications**

If the application gets connected with real banks, it would be nice feature to show push notification on real bank account balance change. It would also remind children to check their savings and maybe add some money to another goal or create a new goal.

### **Design**

The application's design is just a prototyped look right now. It can be easily changed with custom CSS and HTML if needed. To give final look for the production ready application it would be great if there is a professional designer who can use the ideas from the prototype and redesign the Ionic application.

## 8 References

- [1] Majandus- ja Kommunikatsiooniministeerium, „Kiiralaenuturg - analüüs ja ettepanekud,“ 2014. [Online]. Available: [https://www.mkm.ee/sites/default/files/kiiralanuturg\\_analyys\\_ja\\_ettepanekud.pdf](https://www.mkm.ee/sites/default/files/kiiralanuturg_analyys_ja_ettepanekud.pdf). [Accessed 4 May 2017].
- [2] L. Shin, „The 5 Most Important Money Lessons To Teach Your Kids,“ 15 October 2013. [Online]. Available: <https://www.forbes.com/sites/laurashin/2013/10/15/the-5-most-important-money-lessons-to-teach-your-kids/#43598a8b6826>. [Accessed 4 April 2017].
- [3] S. Richmond, „5 Smart Tips For Children's Financial Literacy,“ 24 March 2015. [Online]. Available: <http://www.investopedia.com/articles/personal-finance/032415/5-smart-tips-childrens-financial-literacy.asp>. [Accessed 7 April 2017].
- [4] L. deCos, Opportunity for banks to utilize gamification as a tool to promote financial education to children, Turku University of Applied Sciences, Turku, 2015.
- [5] R. Patil, „Pros and Cons of Cross-Platform Mobile App Development,“ 13 September 2016. [Online]. Available: <https://www.infoq.com/articles/mobile-cross-platform-app-development>. [Accessed 8 April 2017].
- [6] Ionic, „Ionic Framework Homepage,“ [Online]. Available: <http://ionicframework.com/>. [Accessed 5 April 2017].
- [7] C. Tetreault, „Top JavaScript Frameworks for Enterprise Mobility (Part 1/3),“ 23 July 2015. [Online]. Available: <http://devblog.blackberry.com/2015/07/top-javascript-frameworks-for-enterprise-mobility-part-13/>. [Accessed 5 April 2017].
- [8] Pivotal Software, Inc., „Spring Framework,“ [Online]. Available: <https://spring.io/>. [Accessed 5 April 2017].
- [9] Pivotal Software, Inc., „Spring Boot,“ [Online]. Available: <http://projects.spring.io/spring-boot/>. [Accessed 5 April 2017].
- [10] Pivotal Software, Inc., „Spring Security,“ [Online]. Available: <http://projects.spring.io/spring-security/>. [Accessed 5 April 2017].
- [11] MongoDB, Inc., „What is MongoDB,“ [Online]. Available: <https://www.mongodb.com/what-is-mongodb>. [Accessed 5 April 2017].
- [12] V. Hellström, „PSD2 - the directive that will change banking as we know it,“ 20 March 2017. [Online]. Available: <https://www.evry.com/en/news/articles/psd2-the-directive-that-will-change-banking-as-we-know-it/>. [Accessed 05 May 2017].

## Appendix

### I. Source code

- Tarakas Ionic client - <https://github.com/MagnarAr/tarakas-ionic-client>
- Tarakas application server - <https://github.com/MagnarAr/tarakas-core>

For Android, the Ionic application \*.apk file is provided in Tarakas Ionic client git project (link above).

Currently, iOS Ionic application can only be run through *Xcode* (Apple's own development platform) using source code as Apple does not support any official way building an unsigned application, such as \*.app for Android.

Installation/build notes for both platforms are described in Tarakas Ionic client git project.



## II. Android vs iOS user interface

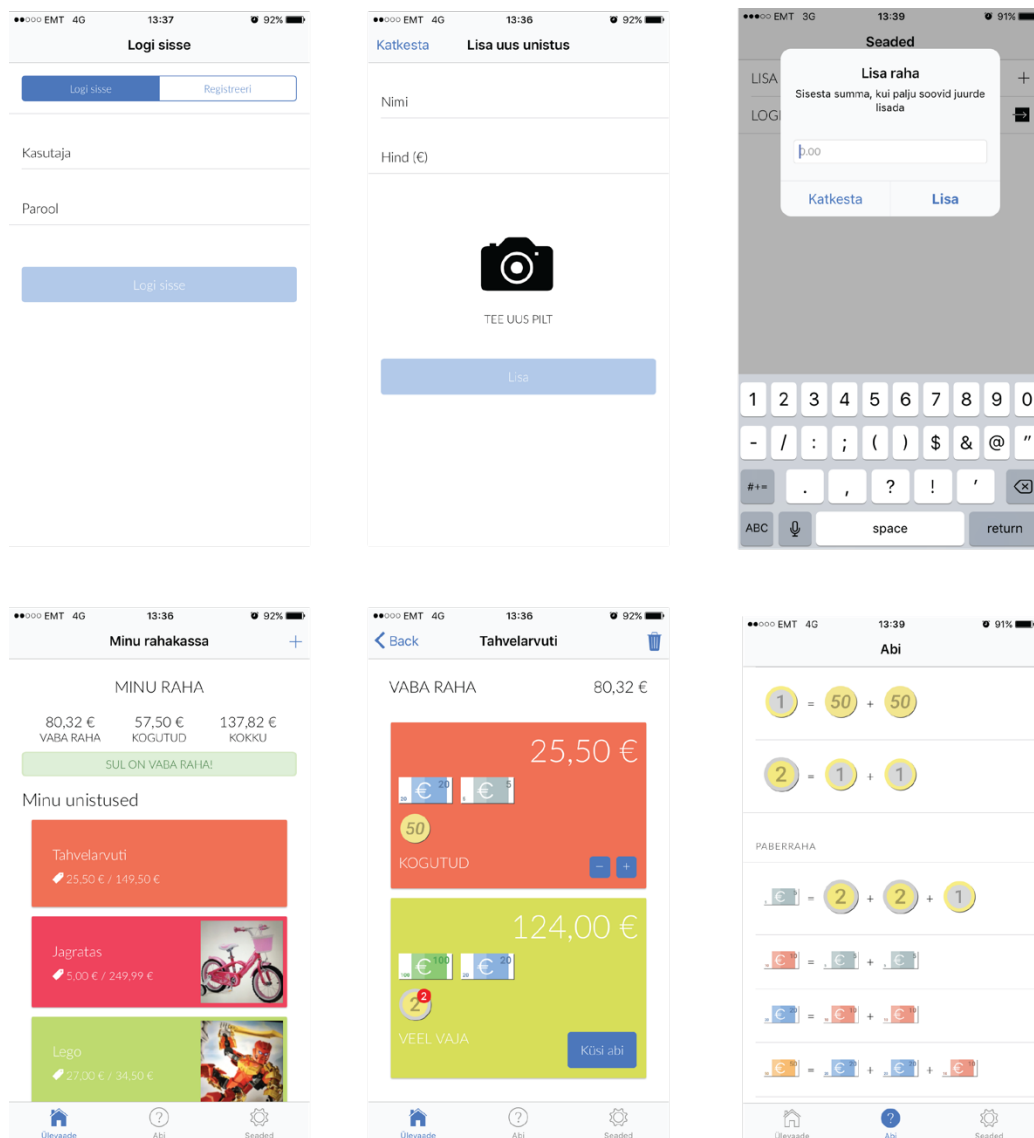


Figure 21. Screenshots of iOS application (iPhone 6S)

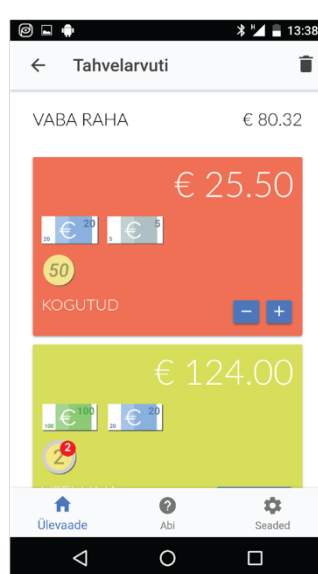
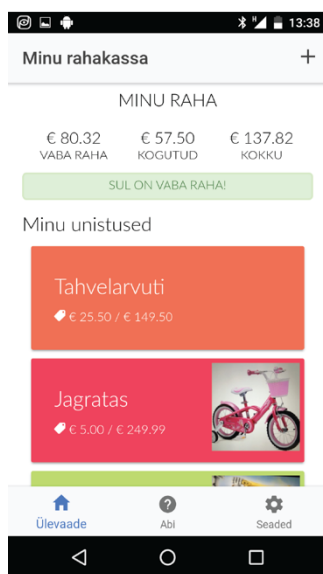
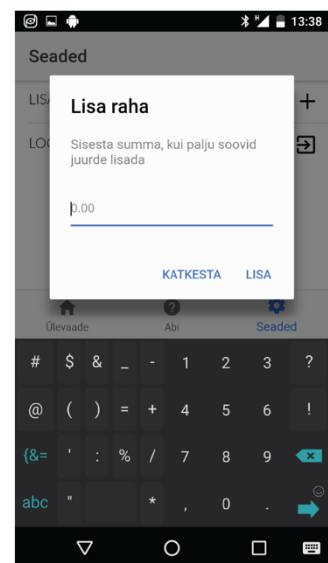
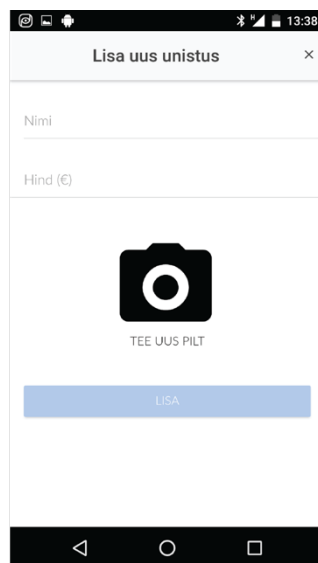
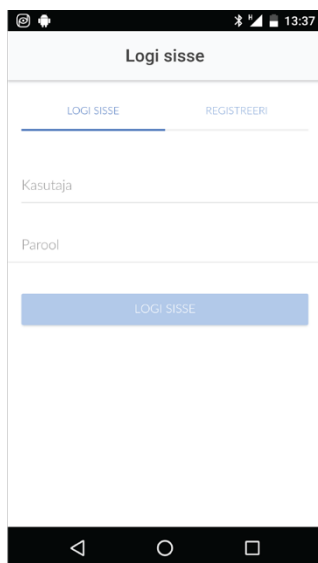


Figure 22. Screenshots of Android application (Nexus 5)

### **III. License**

#### **Non-exclusive licence to reproduce thesis and make thesis public**

I, **Magnar Aruoja**, herewith grant the University of Tartu a free permit (non-exclusive licence) to:

- 1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and
- 1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis **Tarakas – a cross-platform piggy bank application for children**, supervised by Pelle Jakovits and Kristjan Kool,

2. I am aware of the fact that the author retains these rights.
3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **11.05.2017**